AD-A267 151

ENTATION PAGE

PORT DATE	3. REPORT TYPE AND DATES COVERED FINAL/01 JAN 90 TO 31 DEC 92

4. TITLE AND SUBTITLE
EFFICIENT ALGORITHMIC TECHNIQUES FOR COMBINATORIAL PROBLEM (REVISED)

5. FUNDING NUMBERS
2304/DS
AFOSR-90-0107
61102F

6. AUTHOR(S)
PROFESSOR ATALLAH

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
PURDUE RESEARCH FOUNDATION
WEST LAFAYETTE IN 47907-0199
BLACKSBURG, VA 24061

8. PERFORMING ORGANIZATION REPORT NUMBER
AFOSR-TR- 93 0474

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR/NM
110 DUNCAN AVE, SUTE B115
BOLLING AFB DC 20332-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFOSR-90-0107

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE: DISTRIBUTION IS UNLIMITED

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Important algorithms for parallel computational geometry and for string recognition have been developed. These algorithms have been developed for application in a parallel and stochastic setting.

93-16874

DTIC ELECTE JUL 27 1993 S B D

14. SUBJECT TERMS	15. NUMBER OF PAGES
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR(SAME AS REPORT)

NSN 7540-01-280-5500

Standard Form 298 (Rev 2-89)
Prescribed by ANSI Std. 239-18
298-102

Contract AFOSR-90-0107

"Efficient Algorithmic Techniques for Combinatorial Problems"

by M.J. Atallah, A. Apostolico and W. Szpankowski

Report

CONTENTS OF REPORT

A. STATISTICS

While supported by this contract, we have submitted over 55 refereed papers. More than 40 were already published or are in press in leading scientific journals and proceedings. We have also prepared two book chapters, and we are working on three more book chapters. We have supported 5 Ph.D students, four of whom have already graduated. Finally, we presented our results at numerous conferences and workshops, and we delivered over 20 invited talks.

B. RESULTS

We have structured the presentation of our results along three lines: accomplishments, most significant papers, and presentations.

B1. Accomplishments

The primary objective and scope of the project was the development of efficient algorithms for combinatorial problems (in particular geometric, graph, and string problems). We have discovered *general* paradigms that solve a large number of problems. The project

has made a major impact on the important area of parallel computational geometry, algorithmics on strings and probabilistic analysis of algorithms.

Interestingly, in the parallel algorithms area, it was rarely the case that the optimal parallel technique we discovered resembled the previously known sequential method for solving the problem. The implication is that, as far as designing *optimal* parallel algorithms, starting with the existing sequential code and trying to "parallelize it" is often a losing proposition (i.e., one has to start from scratch). We stress the word "optimal", since one might achieve *reasonable* speedup by starting with the sequential code and trying to modify it, and this is in fact an important practical activity in view of the billions of dollars' worth of existing sequential code. But it is important to realize that starting with the sequential code more often than not leads to a suboptimal parallel solution.

In the area of algorithms on strings we proposed the best possible sequential and parallel algorithms for a class of string searching problems. These algorithms were also designed from the practical view point, that is, in a probabilistic framework (see Section C below). We accomplished this in a unified way by suggesting a technique, named "string-ruler approach", which allows to analyze suffix trees in a general probabilistic framework.

We have also solved open problems in stochastic combinatorial optimization, including: the large deviation estimate of a number of jobs in a multiprocessor systems, the maximum size of the hashing with lazy deletion, the length of the compressed word in the Lempel-Ziv compression scheme (which we reduced to a problem on suffix trees). The first problem requires a solution of the long standing open problem of Feller. The second problem was suggested by Van Wyk and Vitter. The last problem solves a conjecture posed by Wyner and Ziv.

B2. Significant Publications

The publications resulting from this contract (some of which are discussed in more detail below) have appeared in established archival journals and conferences (see Section A above). They are extensively referenced in the literature; the references to them in journals and conference papers are too many to enumerate, so we only mention some of the textbooks where they are referenced. M. Atallah is referenced in Miller and Stout (MIT Press), JaJa (Computer Science Press), Leighton (Morgan Kaufmann Publishers), Reif (Morgan Kaufmann Publishers). A. Apostolico is referenced in S. Baase (Addison-Wesley), Van Leuwen (Elsevier), Gonnet and Baeza-Yates (Addison-Wesley), Manber (Addison-Wesley) and others. W. Szpankowski is cited in Gonnet and Baeza-Yates (Addison-Wesley), and Mahmoud (John Wiley and Sons). Often, substantial technical material in these textbooks (sometimes a whole chapter) is based on this work, thus assigning to these contributions a value of significant and lasting impact.

Below, we discuss in more details some of the papers that have been prepared and published during the course of this contract:

1 .**M. Atallah**, P. Jacquet and **W. Szpankowski**, Pattern Matching With Mismatches: A Randomized Algorithm and its Analysis, *Proc. Combinatorial Pattern Matching*, Tucson, 27-40, 1992; Extended version Purdue University TR-92-020, 1992 (submitted to a journal).

This paper solves the following problem: Given a text of length n and a pattern of length m over some (possibly unbounded) alphabet, we consider the problem of finding all positions in the text at which the pattern "almost occurs". Here by "almost occurs" we mean that at least

Table 1: Simulation results for uniform alphabet with $n = 2000$, $m = 400$ and $V = 100$.

L	$\rho = 0.6$		$\rho = 0.7$		$\rho = 0.8$		$\rho = 0.9$	
	%Exact	%Estim.	%Exact	%Estim.	%Exact	%Estim.	%Exact	%Estim.
5	69.75	70.25	75.25	75.75	83.75	84.00	90.50	90.00
10	69.75	69.25	75.25	75.50	83.75	83.50	90.50	90.50
20	69.75	69.50	75.25	75.25	83.75	83.25	90.50	90.50
30	69.75	70.00	75.25	75.00	83.75	83.25	90.50	90.00
40	69.75	70.00	75.25	75.25	83.50	83.50	90.50	90.50
50	69.75	69.75	75.25	75.25	83.75	83.50	90.50	90.25

some fixed fraction ρ of the characters of the pattern (for example, $\geq 60\%$ of them) are equal to their corresponding characters in the text. We design a randomized algorithm that has $O(n \log m)$ worst-case time complexity and computes with high probability all of the almost-occurrences of the pattern in the text. This algorithm assumes that the fraction ρ is given as part of its input, and it works well even for relatively small values of ρ. It makes no assumptions about the probabilistic characteristics of the input. However, a probabilistic analysis of this problem provides values of ρ corresponding to the intuitive notion of similarity between pattern and text. The techniques extend to higher dimensions, with the potential of an important practical impact in the area of vision and image processing. In short, we have a probabilistic proof and experimental results that confirm our belief that the algorithm will perform very well in practice. Table 1 compares %Estim, the value returned by our algorithm, to the exact value %Exact.

2. **A. Apostolico**, Efficient CRCW-PRAM Algorithms for Universal String Searching, *Theoretical Computer Science*, accepted. **A. Apostolico**, Optimal Parallel Detection of Squares in Strings, *Algorithmica*, in press (1992).

We have considered the problem of finding *all* the occurrences of each one of an arbitrary number of *pattern* strings $y_1, y_2, ..., y_g$ in each one of an arbitrary number of *texts* strings $x_1, x_2, ..., x_h$. A solution is given in [6] that takes time $O(\log(\max\{|y_i|\}))$ on a CRCW PRAM with $\sum_{i=1}^{h} |t_i| + \sum_{i=1}^{g} |p_i|$ processors. An intermediate product of the algorithm is a standard representation for strings that requires $O(\log n)$ time and $O(n \log n)$ total work and space for a string of length n, and has the following property of independent interest: Let w and $\bar{w}$ (possibly, $w = \bar{w}$) be any two strings in their respective standard representations. Then, for any chosen pair of substrings w' and $\bar{w}'$ in w and $\bar{w}$, respectively, a CRCW PRAM with $\max(|w'|, |\bar{w}'|)$ processors will return all the occurrences of the shorter of $(w', \bar{w}')$ in the longer of $(w', \bar{w}')$, in constant time. This result provides a novel and very efficient paradigm in structuring huge sequence data banks for instantaneous access in parallel. The need for such data banks arises in diverse applications, including the implementation of text, picture, signal and, last but not least, biomolecular repositories.

Using similar techniques, optimal parallel algorithms have been developed for testing whether a string of n symbols embeds a square, i.e., a substring of the form ww, and for detecting all squares in a string. Studies on squares and other regularities in strings date back to the pioneering work of A. Thue at the beginning of this century - thus mingling with the very birth of theoretical computer science– and they are relevant to many domains of science including systems theory, combinatorics, etc. Also these algorithms run on the CRCW PRAM model of computation. The test takes $O(\log n)$ time and $n/\log n$ or n processors, respectively, depending on whether the alphabet size is bounded or unbounded [2]. The detection of all squares takes $O(\log n)$ time with n processors [3].

All of these algorithms achieve optimum *speedup* (i.e., time times number-of-processors) with respect to the best sequential algorithms available, and they translate into novel serial algorithms

that do not resemble any of their predecessors.

The criteria that subtend all algorithms in the collection of these papers make crucial use of an order relation arbitrarily defined on the input alphabet, along with the induced lexicographic order. Thus, it is shown that assuming an arbitrary alphabet order may lead to discover more efficient solutions to problems on strings which do not imply any notion of such an order. Such a feature is particularly intriguing in the realm of string processing: some lower bounds (e.g., for string editing and longest common sequence problems) hold in a model of computation where character comparisons give outcomes only in [equal, unequal], but the same bounds are not known to hold outside this model.

3. **A. Apostolico, M.J. Atallah**, L. Larmore and H.S. McFaddin, Efficient Parallel Algorithms for String Editing and Related Problems, *SIAM Journal on Computing* 19, 5, 968-988, (1990). **M.J. Atallah** and H.S. McFaddin, Sequence Comparison on the Connection Machine, *Concurrency: Practice and Experience*, 3, 89-107, (1991).

The string editing problem for input strings x and y consists of transforming x into y by performing a series of weighted edit operations on x of overall minimum cost. An edit operation on x can be the deletion of a symbol from x, the insertion of a symbol in x, or the substitution of a symbol of x with another symbol. In various ways and forms, the string editing problem arises in many applications, notably, in text editing, speech recognition, machine vision and, last but not least, molecular sequence comparison. For this reason, this problem has been studied rather extensively in the past, and forms the object of several papers. Our paper gives the best parallel algorithm for this problem for the CREW-PRAM. (The best CRCW-PRAM bound was later given by Atallah in a paper that will soon appear in *Algorithmica*.) More importantly, the techniques used in this parallel algorithm were used to solve a large number of problems apparently unrelated to string editing (for example, computing in parallel shortest paths in the presence of obstacles, as was shown in a recent paper by Atallah and Chen — but there are literally hundreds of applications to these kinds of matrix searching problems, as was pointed out by Aggarwal and Park, who arrived at similar CREW bounds from a different direction and using different techniques). But even the single problem of string editing is of extreme practical importance. In fact it was recently pointed out to us by Jeff Uhlmann (Naval Research Lab, Washington, D.C.) that string editing can be used to solve the all-important *data entry detection problem* (Uhlmann stated that this is a huge problem for many Government and industry databases, one that most people have encountered when using on-line library system and finding multiple entries for the same item that differ only due to a typographical error in one). We supplied Naval Research Labs with our software for solving this problem on the Connection Machine 2 (more on this later). Needless to say that our solution to the general string editing problem implies similar bounds for all the special cases of this problem (like longest common subsequence and approximate pattern matching).

4. **W. Szpankowski**, Asymptotic properties of data compression and suffix trees, *IEEE Trans. Information Theory*, to appear (also presented as extended abstracts in *Data Compression Conference*, Snowbird, 1991, and *Third Symposium on Discrete Algorithms*, Orlando 1992); and **W. Szpankowski**, A generalized suffix tree and its (un)expected asymptotic behaviors, *SIAM J. Computing*, to appear (also presented *Combinatorial Pattern Matching*, Tucson, 1992)

These two papers make major contribution to data compression and understanding of typical behaviors of suffix trees, hence also numerous algorithms on words. For example, the first paper

settles in the negative a conjecture of Wyner and Ziv concerning the behavior of universal data compression scheme. The second paper extends the analysis of the former one and provides further evidence regarding suffix trees. In short, we show that the length of the longest, shortest and typical repeated pattern is $O(\log n)$ where n is the length of a sequence (in fact, much stronger results are reported in these papers). In particular, we show that a typical suffix tree is well balanced, and therefore often brute force algorithms on words are asymptotically optimal on average. In the second paper cited above, we have introduced a new family of suffix trees, called b-suffix trees, that unify in a natural way noncompact suffix trees and compact suffix trees. More precisely, $b = 1$ coincides with noncompact suffix trees, and $b \to \infty$ as $n \to \infty$ corresponds to compact suffix trees. Despite theoretical nature of these results, both papers give background for a new off-line data compression schemes that are asymptotically optimal. We describe below in Section D two new problems that can be attacked by the methodology introduced in these papers.

C. PRODUCTS

We divide this section into the following categories: honors, software prototypes and transitions.

C1. Honors

The quality of the work produced and the recognition it received in the community has resulted in Atallah having editorial duties in seven journals (whereas he had such duties with only 2 journals in 1989), and in serving on many conferences program committees, most recently as Program Committee Co-Chair for the 1992 IEEE Symposium on Parallel and Distributed Processing. Atallah was Invited Speaker to the 1991 Workshop on Parallel Algorithms (New Orleans, Louisiana). A. Apostolico was invited to deliver invited lectures at the first (1990) and second (1991) International Schools on Combinatorial Pattern Matching held in Paris, France (A. Aho, M. Crochemore and Z. Galil, co-lecturers), and London (M. Crochemore and Z. Galil, co-lecturers). He served on the Program Committee of the 1992 Combinatorial Pattern Matching Conference in Tucson, AZ, and is chairing the Program Committee of this Conference for 1993. Apostolico also directed an international school on String Algorithmics and its Applications held at the Fibonacci Institute in Trento, Italy, in 1990 (A. Ehrenfeucht and M. Zuker, co-lecturers), served on various other Program Committees, edited a special Issue of Algorithmica on "String Algorithmics" due to appear in 1993, and gave invited presentations at various institutions and conferences. W. Szpankowski was guest editor in the *IEEE Trans. on Automatic Control.* He gave several invited talks, including one at Boston during the IMS Workshop "Direction in Applied Probability". He was also awarded a one year stay at INRIA, France to continue his research on analytical and probabilistic analyses of algorithms.

C2. Software Prototypes

Some of our published algorithms have been implemented on commercially available parallel hardware: string editing on Alliant and Connection Machine, jointly with Purdue graduate student Scott McFaddin. (We also learned that some of our parallel computational geometry algorithms were implemented on the Connection Machine by Guy Belloch, currently at CMU — Blelloch is not connected to the Contract or the PIs.) The experimental work done jointly with McFaddin took place during a one-month visit to the RIACS Institute's Center for Advanced Architectures, at the NASA Ames Research Center, and later

by remote access to NASA Ames' Connection Machine 2. We developed and implemented parallel algorithms for solving the string editing problem on the Connection Machine 2, and were able to achieve spectacular speedups. These results appeared in a recent issue of *Concurrency: Practice and Experience* (1991).

We have also implemented optimal and heuristic algorithms for the exact bipartite matching (i.e., linear assignment problem) and the bottleneck optimization problems (e.g., bottleneck assignment problem, bottleneck location problem, etc.). We have obtained significant improvement in the running time, that was theoretically predicted.

C3. Transitions

The above-mentioned software we developed for parallel string editing on the Connection Machine 2 was requested by Jeff Uhlmann (Naval Research Lab, email: uhlmann@nrl-5570-gw.itd.nrl.navy.mil). We supplied it to him (free of charge, of course). The practical problem he wanted to use it on is the *data entry detection problem.* Uhlmann stated that this is a huge problem for many Government and industry databases, one that most people have encountered when using on-line library system and finding multiple entries for the same item that differ only due to a typographical error in one.

D. NEW RESEARCH DIRECTIONS

Below are some of the new research directions we plan on pursuing. We would be happy to explore these and other ideas for future research in a new, detailed proposal.

D1. Military Applications of CG and String Matching

A new direction we are taking, at the suggestion of Jeff Uhlamnn from NRL, is to explore the use of computational geometry (CG) techniques for solving problems that arise in simulations for SDI and other military applications. The techniques of computational geometry should have direct applicability to such problems. As an example of such a problem, consider flying objects each of which has a zone of influence surrounding it: one can use CG techniques for answering various queries about these (intersection queries, for example, where an intersection between two zones of influence could mean the possibility for interaction between the two corresponding objects). The difference with previous work in CG is that, whereas previous work was supply-driven in the sense that research was done in CG that would probably have military applications, the new research is demand-driven in that one starts with problems formulated by the military and then tries to solve them using CG. This will end up enriching the CG area itself, by involving it in new problems that would not have been considered otherwise.

The comments made above for CG can also be made for pattern matching. For example, one could extend pattern matching techniques to real-time observations of multiple sequences of data (one sequence could measure the level of enemy radio communications, another the number of aircraft detected by radar, etc). The idea is that the simultaneous appearances of certain patterns on the various observed sequences could signal something serious (a likely enemy attack, for example). The published string matching techniques are inadequate for this kind of situation, since now the composite "pattern" to be detected simultaneously depends on what happens on more than one input sequences (each sequence gives part of the evidence).

D2. Data Compression

We have begun investigations that are shedding light on the usefulness of current data compression algorithms, and hold the promise of resulting in new, useful algorithms.

In data compression by textual substitution substrings that appear many times in a given textstring are replaced by a small set of pointers to a unique common copy (for instance, by giving (1) a textstring position starting from which the substring can be recopied, and (2) the length of that substring). Disparate conventions, mostly concerning the location of the common copy, and the mechanics of the encoding-decoding process, give rise to many *schemes* for compression.

Unfortunately, it turns out that, with one or two noteworthy exceptions, the optimal implementation of the majority of such schemes translates into NP-complete problems. In view of this, the task should be to find efficient approximate solutions. This task poses interesting questions. To give one example, consider the following *steepest descent* approximation. Given a text w, we find a substring u of the text w yielding the best compression and then replace each of the occurrences of u but one with a pair of pointers to the untouched occurrence of u in w. The process is then repeated on the resulting string, and so on, until a repetition results in little or no compression. The maximum time and space efficiency with which such a paradigm can be implemented is not known. One part of each repetition consists of computing the string statistics without overlap. Thus a probabilistic analysis of the latter problem may lead to interesting estimates about the compression scheme.

Additional questions involve linear time, locally adaptive data compression schemes such as Lempel-Ziv algorithm. In these schemes, the production of a compressed string is interleaved with the construction of the suffix tree for the source string. These schemes enjoy a number of desirable features from the information-theoretic viewpoint. In practice, their performance is often beset by the somewhat subtle interplay between pointer sizes and dictionary parameters (say, number of entries, and average length). Various techniques for the compact encoding of integers in an unbounded domain have been devised that could be used to keep the sizes of pointers to a minimum. However, the central problem is still that of which probabilistic assumptions about the source string would guarantee a good compression by such methods. Using some facts from the analysis of suffix trees and other regularities on strings, we discovered some surprising facts.

Now, we describe two problems that seem to be in the range of our methodology described in our papers [54, 56, 58, 57, 59]. The first one is an *off-line construction* of the optimal Lempel-Ziv algorithm. The second problem deals with the *data compression with errors*. More precisely, in the off-line construction of the optimal data compression scheme, we assume that a transmitter has at its disposal the *whole* string that is to be compressed. Our analysis of suffix trees suggests that we should send first the longest suffix that occured twice in the block, then the next longest that occurred twice, etc. The difficulty is, however, that this off-line construction should parse the block in a manner similar to the one proposed by Lempel and Ziv. We have identified a scheme that should work well for this problem. We would like to implement this algorithm in serial and parallel framework. If LZ_n is the compression factor for the Lempel-Ziv algorithm (i.e., the ratio of the overhead to the average length of the repeated pattern when the dictionary is of size n), then we expect that our off-line data compression scheme will lead to the compression factor

of order $LZ_n - O(\sqrt{\log(\log n)/\log n})$. It is well known that the Lempel-Ziv algorithm is asymptotically optimal, hence one may only expect an improvement of order $o(1)$.

In the second problem, we consider the behavior of the Lempel-Ziv algorithm when the channel may introduce errors; that is, the dictionary sequence in the transmitter and the receiver might *not* be the same. The first question is how much do we lose in such a faulty environment. The data compression ratio is no any longer the entropy of the source. We believe we can answer this question by extending our analysis of suffix trees. The other – more interesting problem – is how to reconstruct the original sequence. This problem resembles the pattern matching with mismatches (approximate matching, etc.) and we believe that we have a strategy to attack this problem. We plan serial and parallel implementations of this algorithm. A solution to this problem will have direct applications in the Air Force mission.

E. PAPERS ACKNOWLEDGING THE CONTRACT

[1] D. Aldous, M. Hofri and W. Szpankowski, Maximum Size of Some Dynamic Data Structures: Hashing with Lazy Deletion Revisited, *SIAM J. Computing*, 21, 4, 1992.

[2] A. Apostolico, Optimal Parallel Detection of Squares in Strings - Part I: Testing Square-freedom, *Algorithmica*, to appear.

[3] A. Apostolico, Optimal Parallel Detection of Squares in Strings - Part II: Detecting all Squares, *Algorithmica*, to appear.

[4] A. Apostolico and A. Ehrenfeucht, Efficient Detection of Quasiperiodicities in Strings, *Theoretical Computer Science*, to appear.

[5] A. Apostolico, M. P. Farach and C. S. Iliopoulos, Optimal Superprimitivity Testing for Strings, *Information Processing Letters*, 39, 17-20, 1991.

[6] A. Apostolico, Efficient CRCW-PRAM Algorithms for Universal Substring Searching, *Theoretical Computer Science*, to appear.

[7] A. Apostolico, D. Breslauer and Z. Galil, Optimal Parallel Algorithms for Periods, Palindromes and Squares, *Proceedings of 1992 ICALP*, Springer-Verlag LNCS, in press.

[8] A. Apostolico, S. Browne and C. Guerra, Fast Linear-Space computations of Longest Common Subsequences, *Theoretical Computer Science*, Special Issue on Combinatorial Pattern Matching, 92, 3-17, 1992.

[9] A. Apostolico, G. Italiano, G. Gambosi and M. Talamo, The Union Find with Unlimited Backtracking, Purdue CSD-TR 908, submitted to a journal.

[10] A. Apostolico and M. Crochemore, Optimal Canonization of All Substrings of a String, *Information and Computation*, 95, 76-95 1991.

[11] A. Apostolico and M. Crochemore, Fast Parallel Lyndon Factorization With Applications, Purdue CSD-TR 932, submitted to a journal.

[12] A. Apostolico, M. Atallah, L. Larmore and H.S. McFaddin, Efficient parallel algorithms for string editing and related problems, *SIAM Journal on Computing* 19, 5, 968-988, (1990).

[13] A. Apostolico, M. Atallah and S.E. Hambrusch, New Clique and Independent Set Algorithms for Circle Graphs, *Discrete Applied Mathematics*, in press.

[14] A. Apostolico and F.P. Preparata, Data Structures and Algorithms for the String Statistics Problem, Purdue CSD-TR 572, submitted to a journal.

[15] A. Apostolico and W. Szpankowski, Self-alignments in Words and their Applications, *J. of Algorithms*, 13, 1992.

[16] M. J. Atallah, S. E. Hambrusch, and L. E. TeWinkel, "Parallel Topological Sorting of Features in a Binary Image," to appear in an *Algorithmica* Special Issue on Parallel Algorithms for Geometric Problems on Digitized Pictures.

[17] M. J. Atallah, D. Z. Chen and H. Wagener "An Optimal Parallel Algorithm for the Visibility of a Simple Polygon from a Point," *J. of the ACM* (in press).

[18] M. J. Atallah and J-J. Tsay, "On the Parallel-Decomposability of Geometric Problems," to appear in *Algorithmica* (1992).

[19] M. J. Atallah and H. S. McFaddin, "Sequence Comparison on the Connection Machine," *Concurrency: Practice and Experience* **B3** (1991).

[20] M. J. Atallah, C. Ribeiro and S. Lifschitz, "A Linear Time Algorithm for the Computation of Some Distance Functions between Convex Polygons," accepted for publication in *RAIRO J. Oper. Res.*

[21] M. J. Atallah, C. Ribeiro and S. Lifschitz, "Computing Some Distance Functions Between Polygons," accepted for publication in *Pattern Recognition.*

[22] M. J. Atallah, "A Faster Parallel Algorithm for a Matrix Searching Problem," accepted for publication in *Algorithmica.*

[23] M. J. Atallah and D. Z. Chen, "Parallel Rectilinear Shortest Paths with Rectangular Obstacles," accepted for publication in *Computational Geometry: Theory and Applications.*

[24] M. J. Atallah and S. R. Kosaraju, "An Efficient Parallel Algorithm for the Row Minima of a Totally Monotone Matrix," *Proc. 2d ACM-SIAM Symp. on Discrete Algorithms*, San Francisco, California, 1991, pp. 394-403. (Accepted for publication in *J. of Algorithms.*)

[25] M. J. Atallah, "Parallel Techniques for Computational Geometry," accepted for publication in *Proc. of IEEE.*

[26] M. J. Atallah, C. Lock, D.C. Marinescu, H.J. Siegel, and T.L. Casavant, "Co-Scheduling Compute-Intensive Tasks on a Network of Workstations: Models and Algorithms," *Proc. 11th Int. Conf. on Distributed Computing Systems*, Arlington, Texas, 1991.

[27] M. J. Atallah, Frank Dehne, Russ Miller, Andrew Rau-Chaplin, and Jyh-Jong Tsay, "Multisearch Techniques for Implementing Data Structures on a Mesh-Connected Computer," *Proc. 2d Annual ACM Symp. on Parallel Algorithms and Architectures*, Hilton Head, South Carolina, 1991.

[28] M. J. Atallah, M. T. Goodrich, and S. R. Kosaraju, "On the Parallel Complexity of Evaluating Some Sequences of Set Manipulation Operations," submitted to a journal.

[29] M. Atallah and P. Jacquet and W. Szpankowski, Pattern matching with mismatches: A probabilistic analysis and a randomized algorithm, *Proc. Combinatorial Pattern Matching*, Tucson, 1992, pp. 27-40.

[30] M. Atallah and P. Jacquet and W. Szpankowski, A Probabilistic Approach to Pattern Matching with Mismatches, Purdue University, CSD-TR-1007 1990; submitted to a journal.

[31] M. J. Atallah, P. Callahan, and M.T. Goodrich, "P-Complete Geometric Problems," submitted to a journal.

[32] M. J. Atallah and J. B. Manning, "A Linear Time Algorithm for Computing the Symmetries of Embedded Planar Graphs," in preparation.

[33] M. J. Atallah and M. T. Goodrich, "Concurrent Conflict-Free Searching of Parallel Data Structures," in preparation.

[34] L. Devroye, W. Szpankowski and B. Rais, A Note on the Height of Suffix Trees, *SIAM J. Computing*, 21, 48-53, 1992.

[35] L. Georgiadis and W. Szpankowski, Stability of Token Passing Rings, *Queueing Systems*, 1992, in press.

[36] M. T. Goodrich, M. J. Atallah, and M. H. Overmars "Output-Sensitive Hidden Surface Elimination for Rectangles," accepted for publication in *Info. and Computation.*

[37] P. Jacquet and W. Szpankowski, Autocorrelation on Words and Its Applications: Analysis of Suffix Trees by String-Ruler Approach, INRIA TR-1106, revision in *J. Combinatorial Theory. Ser. A.*

[38] P. Jacquet and W. Szpankowski, Analysis of Digital Tries with Markovian Dependency, *IEEE Trans. Information Theory*, 37, 1470-1475, 1991.

[39] P. Jacquet and W. Szpankowski, What can we learn about suffix trees from independent tries ? *1991 Workshop on Algorithms and Data Structures*, Lecture Notes in Computer Science, 519 (eds. F. Dehne, J. Sack and N. Santoro), Springer-Verlag 1991, pp. 228–229.

[40] P. Kirschenhofer, H. Prodinger and W. Szpankowski, Digital Trees Again Revisited: The Internal Path Length Perspective, Purdue University, CSD-TR-989, 1990; submitted to a journal.

[41] P. Kirschenhofer, H. Prodinger and W. Szpankowski, Multidimensional Digital Searching and Some New Parameters in Tries, Purdue CSD-TR-052, 1991; submitted to a journal.

[42] P. Kirschenhofer, H. Prodinger and W. Szpankowski, How to count quickly and accurately: A unified analysis of probabilistic counting and other related problems, *International Conference on Automata, Languages, and Programming (ICALP'92)*, Vienna, 1992.

[43] G. Louchard and W. Szpankowski, String Matching: A Preliminary Probabilistic Analysis, Universite de Bruxells, TR-217, 1991.

[44] B. Rais, P. Jacquet and W. Szpankowski, A Limiting Distribution for the Depth in PATRICIA Tries, *SIAM J. Discrete Mathematics*, 1992, in press.

[45] B. Rais, P. Jacquet, and W. Szpankowski, Typical Behavior of Patricia Tries, *28-th Annual Allerton Conference on Communication, Control and Computing*, University of Illinois at Urbana-Champaign, pp. 924-925, 1990.

[46] J. Sadowsky and W. Szpankowski, The Probability of Large Queue Lengths and Waiting Times in a Heterogeneous G|G|c Queue, submitted to a journal (revised in 1991).

[47] J. Sadowsky and W. Szpankowski, Maximum queue length and waiting time revisited: Multiserver G|G|c queues, *Probability in the Engineering and Informational Science*, 6, 157-170, 1992.

[48] W. Szpankowski, A Characterization of Digital Search Trees from the Successful Search Viewpoint, *Theoretical Computer Science*, 85, 117-134, 1991.

[49] W. Szpankowski, On the Height of Digital Trees and Related Problems, *Algorithmica*, 6, pp. 256-277, 1991.

[50] W. Szpankowski, Patricia Tries Again Revisited, *Journal of the ACM*, 37, pp. 691-711, 1990.

[51] W. Szpankowski, Combinatorial optimization through order statistics, *Second Annual International Symposium on Algorithms*, Taiwan, 1991.

[52] W. Szpankowski, Towards Stability Criteria for Multidimensional Distributed Systems: The Buffered ALOHA Case, Purdue University, CSD-TR-983, revised 1991; submitted to a journal.

[53] W. Szpankowski, Another Unified Approach to Some Bottleneck and Capacity Optimization Problems, Purdue University, CSD-TR-1022, 1990; submitted to a journal (revised in 1991).

[54] W. Szpankowski, Asymptotic Properties of Data Compression and Suffix Trees, *IEEE Trans. on Information Theory*, to appear.

[55] W. Szpankowski, On Some Combinatorial Optimization: Bottleneck and Capacity Assignment Problems, *28-th Annual Allerton Conference on Communication, Control and Computing*, University of Illinois at Urbana-Champaign, pp. 92-101, 1990.

[56] W. Szpankowski A Typical Behavior of Some Data Compression Schemes, *Proc. of Data Compression Conference*, Snowbird, pp. 247-256, 1991.

[57] W. Szpankowski, (Un)Expected behavior of typical suffix trees, *Third Annual ACM-SIAM Symposium on Discrete Algorithms*, 1992, pp. 422-431.

[58] W. Szpankowski, A Generalized Suffix Tree and Its (Un)Expected Asymptotic Behavior, *SIAM J. Computing*, to appear.

[59] W. Szpankowski, Probabilistic analysis of generalized suffix trees, *Proc. Combinatorial Pattern Matching*, Tucson, 1992, pp. 1-14.